# A GRAMMATICAL MODEL OF ORGANIZATIONAL ROUTINES IN A TECHNICAL SERVICE ORGANIZATION

by

Brian T. Pentland & Henry H Rueter

CCS TR # 177, Sloan WP # 3721-94

August 1994

# A GRAMMATICAL MODEL OF ORGANIZATIONAL ROUTINES

# IN A TECHNICAL SERVICE ORGANIZATION

DRAFT
Accepted for Publication in
Administrative Science Quarterly

Brian T. Pentland
University of California, Los Angeles

Henry H. Rueter
University of Michigan

The authors wish to thank Stephen Barley, Michael Cohen, Connie Gersick, Barbara Lawrence, Lance Sandelands, Robert Sutton, Burt Swanson, Karl Weick, Elaine Yakura and the anonymous ASQ reviewers for their comments on earlier drafts of this paper. We would also like to thank Linda Pike for her expert editing, and Louise Soe and Sidne Ward for their help with the data analysis. An earlier version of this paper was presented at the Academy of Management, Las Vegas, NV, August 1992.

# A GRAMMATICAL MODEL OF ORGANIZATIONAL ROUTINES IN A TECHNICAL SERVICE ORGANIZATION


## ABSTRACT

This paper explores the sequential structure of work processes in a task unit whose work involves high numbe ̦ of exceptions, low analyzability of search, frequent interruptions and extensive deliberation, and cannot be characterized as routine under any traditional definition. Yet a detailed analysis of the sequential pattern of action in a sample of 168 service interactions reveals that most calls follow a repetitive, functionally similar pattern. This apparent contradiction presents a challenge to our theoretical understanding of routines: how can apparently non-routine work display such a high degree of regularity? To answer this question, we propose a new definition of organizational routines as a set of functionally similar patterns and a illustrate a new methodology for studying the sequential structure of work processes using rule-based grammatical models. This approach to organizational routines juxtaposes the structural features of the organization against the reflective agency of organizational members. Members enact specific performances from among a constrained (but potentially large) set of possibilities that can be described by a grammar, giving rise to the regular patterns of action we label routines.

Routines occupy a critical position in organization theory. As genetic material, routines are used to explain the inertial quality of organizational structure in evolutionary theories (Nelson & Winter, 1982; Hannan & Freeman, 1983). As memory, routines have become a cornerstone in theories of organizational learning and adaptation, as well (Levitt & March, 1988; Cohen, 1991; Walsh & Ungson, 1991). Routines occupy the crucial nexus between structure and action, between the organization as an object and organizing as a process. In spite of their importance to organizational theory, there have been very few attempts to deal empirically with the sequential structure of organizational routines. Instead, existing research has usually collected summary measures that ignore the sequential structure of the action (Lynch, 1974; Daft and MacIntosh, 1981; Withey, Daft, and Cooper, 1983).

Routines are difficult to study because they are essentially complex patterns of social action. Our tools for characterizing the variable properties of static objects are well developed, but our tools for characterizing the sequential structure of patterns of action are not. An additional difficulty lies in two different senses of the word "routine." As a noun, "routine" is used to objectify a collective capacity to perform recognizable patterns of action (Nelson and Winter, 1982). The patterns themselves may vary considerably from instance to instance, so a label is needed to summarize them as a fixed entity. As an adjective, "routine" indicates a judgment about a variable property of a pattern of action, such as repetitiveness (Gersick and Hackman, 1990), analyzability (Perrow, 1967), variety (Daft and MacIntosh, 1981), or even "mindlessness" (Ashforth and Fried, 1988). These two senses of the term seem to lead to a considerable amount of confusion about the underlying construct. Perhaps this is why routines are usually spoken of vaguely, in the plural, rather than concretely, in the singular.

The purpose of this paper is to present a theoretical model that juxtaposes our concept of routines as patterns of action with our concept of routineness as a property that such patterns

might possess. We illustrate this model with an example, taken from the software support function of a medium sized software vendor, that presents us with an apparent contradiction. This task unit exhibits repetitive, functionally similar patterns of action in response to defined stimuli (classic features of organizational routines), yet has high task variety, low task analyzability, and a highly deliberative search process filled with interruptions (classic features of non-routine work).

To resolve this apparent contradiction, we believe it is necessary to rethink the concept of routines and the methods with which we study them. In addition to clarifying the distinction between routines as patterns of action and routineness as a variable property of such patterns, we believe it is necessary to examine the sequential structure of the patterns themselves. To do so, we suggest a novel methodology for representing routines using grammatical models (Pentland, forthcoming; Olson, Herbsleb and Rueter, forthcoming). This paper builds on our prior work by applying the grammatical modeling technique to sequential data from a specific organization. There are many ways to model sequences of events (Abbott, 1991), but grammatical models have distinct advantages when applied to organizational routines and other repetitive organizational processes. First, a grammar does not specify a fixed outcome; it defines a set of possibilities from among which members accomplish specific sequences of action. For this reason, grammatical models acknowledge both structure and agency in a way that is highly congenial to the theoretical conceptions of routines mentioned above. Second, grammatical models can capture the layered quality of action in organizations, where routines often have "sub-routines" that can be rearranged and combined to form new routines (March and Simon, 1958). Third, grammatical models are well suited to representing dependencies between events that may be widely separated in an observed sequence (Chomsky, 1956), which are commonplace in organizational routines. Finally, grammatical models have potential practical value because they provide a clear way to distinguish normatively correct instances of a routine from other instances. Each of these features will be demonstrated in the analysis that follows.

We begin by examining various concepts of routines in the organizational and sociological literature. Drawing on these perspectives, we argue that an organizational routine can be seen as a set of possible patterns that need not be fixed or automatic. Our argument parallels that of Simon (1969) in some respects, in that we argue that situational structures can generate complex patterns of behavior. To demonstrate our argument, we use the example of a task unit that violates the traditional definitions of routineness from the organizational literature. We then use a new grammatical method (Olson et al, in press) to analyze the sequential patterns of action in this task unit to show that in spite of its apparent non-routineness, it exhibits a high degree of sequential structure as defined by the grammar. These findings reinforce the need for a new, broader formulation of the concept of organizational routines, and a new set of tools for analyzing their sequential structure.

## DEFINITIONS OF ORGANIZATIONAL ROUTINES

At the risk of oversimplifying a large, diverse body of work, one can discern some broad perspectives in the literature on routinized behavior. There is, for example, a micro-level perspective that emphasizes the cognitive processes of individuals (Ashforth and Fried, 1988), and a more macro-level perspective that emphasizes structural and institutional constraints (Nelson and Winter, 1982). Levels of analysis are blurred, of course, because situational factors influence script selection and performance (Gioia and Poole, 1984), and the cognitive processes of individuals can enact situational structures (Weick, 1979). There is also a difference of opinion as to whether routines are fixed entities that exist independently of their accomplishment (March and Simon, 1958) or as the performances of acting subjects that unfold anew each time (Giddens, 1984). The difference is similar to Weick's (1979) distinction between organization and organizing; the noun form encourages us to see the phenomenon as a static thing, while the verb form encourages us to see the phenomenon as a dynamic process. And as mentioned above, "routineness" is a property that such patterns might possess, whether they are fixed or dynamic. In the discussion that follows, we distill some

insights from each of these themes and introduce a novel, grammatical perspective on organizational routines.

**Routines as Automatic Responses**

Early concepts of routines emphasized a fixed pattern of individual behavior in response to a defined stimulus. Using the newly popular computational metaphor, March & Simon (1958: 142) asserted that "[m]ost behavior, and particularly most behavior in organizations, is governed by performance programs." March & Simon (1958) went on to consider when an activity could be considered "routinized":

> We will regard a set of activities as routinized, then, to the degree that choice has been simplified by the development of a fixed response to defined stimuli. If search has been eliminated, but a choice remains in the form of a clearly defined and systematic computing routine, we will still say that the activities are routinized. (March & Simon, 1958: 142)

The idea of a fixed response to a given stimulus, accompanied by the absence of search, provides the foundation upon which much subsequent theorizing about routines has been based. Throughout their discussion, March and Simon (1958) argue that the link between the stimulus and the response lies in the perceptions and reactions of the individuals in the organization.

More recently, Ashforth and Fried (1988) use the concept of an event schema or script (Schank and Abelson, 1977) to argue that routines are sustained by the cognitive structures of individual organizational members. Ashforth and Fried (1988) argue that scripted behavior occurs under the following circumstances: (1) presence of an event schema; (2) categorizable stimulus cues; (3) presence of action rules; (4) minimal required effort; (5) absence of unstructured subroutines; and (6) absence of interruptions and unmet expectations. Weiss and Ilgen (1985) offer a similar list of criteria for routinization of individual behavior. Given that these conditions appear to be rather common, Ashforth and Fried (1988) argue that

scripted or mindless behavior is commonplace in operating routines, decision making, formal and informal interactions, and power-based interactions. In this respect, their argument mirrors that of March and Simon (1958).

At the group level of analysis, Gersick and Hackman (1990: 69) offer a definition of "habitual routines" that retains the notions of a fixed stimulus and the absence of search:

> A habitual routine exists when a group repeatedly exhibits a functionally similar pattern of behavior in a given stimulus situation without explicitly selecting it over alternative ways of behaving.

Unlike Ashforth and Fried (1988), Gersick and Hackman (1990) do not rely directly on the concept of individual scripts as an explanatory construct. They argue that at the group level, habitual routines emerge out of the complex interaction of group members. Routines are repetitive patterns of action that are "functionally similar," but not necessarily fixed. In other words, there may be some variety in the patterns enacted by the group, but no explicit deliberation.

At the organizational level of analysis, Nelson & Winter (1982) introduced a wide variety of metaphors for routines: routines as genes, routines as memory, routines as truce, routines as targets for control, replication, and imitation. Each of these metaphors portrays a routine as a kind of thing. The definition of routine offered by Winter (1986) expands on March & Simon's (1958) notion of a fixed response to include patterns of behavior that function as a recognizable unit:

> ...to the extent to which that reference is made to a relatively complex pattern of behavior (or the theoretical representation of such a pattern) triggered by a relatively small number of initiating signals or choices and functioning as a recognizable unit in a relatively automatic fashion, to that extent reference is

> made to something called a routine in evolutionary theory. (Winter, 1986: 165)

Instead of focusing on measurable properties of a task unit's work flow (Perrow, 1967), Nelson and Winter (1982) shifted the focus of theoretical interest to the patterns themselves, objectified under the label "routines." The definition retains the familiar elements of stimulus and automatic response, but unfortunately, the concept of routine as used in evolutionary theory is unacceptably vague and encompassing, as this quote from Winter (1986: 165) illustrates:

> Nelson and I use the word "routine" as the generic term for a way of doing things. It is simultaneously the counterpart of a wide range of terms employed in everyday life and in various theoretical languages, including those of orthodox and behavioral economic theory; among these terms are decision rule, technique, skill, standard operating procedure, management practice, policy, strategy, information system, information structure, program, script and organization form.

Winter's (1986) definition is both troubling and intriguing because it draws upon a wide range of constructs across all levels of analysis. In all the manifestations Winter (1986) alludes to, however, routines are essentially automatic, executed without explicit deliberation or choice.

## Routines as Effortful Accomplishments

In recent social theory, routines are also a central concept, but have received a rather different theoretical treatment. Giddens (1984) identifies routinization as a crucial aspect of the theory of structuration: "The concept of *routinization*, as grounded in practical consciousness, is vital to the theory of structuration" (p. 60, emphasis in original). He goes on to argue that:

> The regular or routine features of encounters, in time as well as in space, represent institutionalized features of social systems. Routine is founded in

> tradition, custom or habit, but it is a major error to suppose that these phenomena need no explanation, that they are simply repetitive forms of behavior carried out 'mindlessly'. On the contrary, as Goffman (together with ethnomethodology) has helped to demonstrate, the routinized character of most social activity is something that has to be 'worked at' continually by those who sustain it in their day-to-day conduct. (Giddens, 1984: 86).

While Giddens does not offer an explicit definition of organizational routines per se, the critical insight here is that routinized social activity is not mindless or automatic, but rather an effortful accomplishment. A similar theme can be found in the literature on discourse and conversation analysis. As Schegloff (1982: 89) argues, interaction must always be treated as "an achievement from among possibilities," not something given or pre-planned. Garfinkel (1967) and his followers have demonstrated repeatedly that the normal troubles and practical contingencies of daily life require constant interpretation and repair (Heritage, 1984).

Even some of the most routinized kinds of encounters, such as fast food service (Leidner, 1993) and buying stamps (Ventola, 1987) exhibit a considerable amount of variety and require effort on the part the participants to accomplish successfully. Customers need to be trained and controlled so that they conform to the expectations of the service provider (Leidner, 1993). Service interactions are especially prone to variation, of course, because they are interactional products. Note that organizational routines frequently require the participation of multiple individuals; as complex interactional products, they are unlikely to unfold the same way every time.

These studies call into serious question any concept of routines as fixed or automatic. One could potentially resolve the apparent disagreement by suggesting that the authors are talking about different phenomena; perhaps what Ashforth and Fried (1988) mean by a routine is different than what Giddens (1984) means. Unfortunately, insofar as either of them uses concrete

examples (neither reports any data), they appear to be talking about the same universe of activity: formal and informal encounters and task situations of all kinds.

The differences in these two broad perspectives on routinized activity are not surprising, given that each perspective emerged as a corrective to an overzealous alternative. In the organizational behavior literature, the idea that routines are automatic was a reaction against the economic dogma that all decision making involves perfect knowledge and optimal choice. Behavioral economics can also be seen as a continuing reaction against this entrenched orthodoxy (Winter, 1986). In sociology (and ethnomethodology, in particular), the idea that routines are effortful accomplishments was a reaction against the Parsonian notion of the social agents as judgmental dopes whose behavior was predetermined by social structure (Heritage, 1984). Seen in the context of their antitheses, each of these theses makes a great deal of sense.

There may also be a methodological bias that can help account for the difference in perspectives. The perspective that views routines as effortful accomplishments usually draws on detailed ethnographic observations of people working and interacting (e. g., Garfinkel, 1967), so it is unavoidable that all of the minor troubles and improvisations of daily life should be apparent. In contrast, the perspective that views routines as automatic responses is associated with methods designed to elicit and then confirm a specific pattern in the minds of subjects (e. g., Leigh and McGraw, 1989). It should not be surprising that these two methodological approaches lead to differing interpretations of the phenomenon.

## ROUTINES AS GRAMMARS OF ACTION

To help reconcile these apparently conflicting perspectives on routinized social action, we propose a theoretical model that allows us to integrate the insights from each of them. Such a model of organizational routines must acknowledge that routinized behavior is constrained and enabled by the cognitive structures of individuals, such as scripts, as well as physical and social

structures of the organization. At the same, it must allow for the individual effort and agency that gives rise to the particular patterns we observe.

We believe that grammatical models are a particularly good choice for this problem, because grammars define sets of possibilities. The concept of grammar can be applied to any ordered sequence of elements or events, linguistic or otherwise. One can create grammars for stories (Prince, 1973; Ryan, 1979; Colby, Kennedy and Milanesi, 1990), tasks (Sandelands, 1987), political history (Alker, 1987), regular polygons (Miclet, 1986), chromosomes (Gonzalez and Thomason, 1978) and nearly anything else that can be described as an ordered set. To the extent that organizational processes (such as routines) can be described as ordered sets of actions, they can also be described by grammars. By expressing the sequences of action observed in an organizational process using a grammar, we can detect sequential structure and functionally similar patterns. Our method builds on Salancik and Leblebici (1988), who used a grammatical model to represent the possible patterns of interaction in food service transactions. We extend on their work by using a grammatical model to express the sequential structure of work processes as observed in a particular organization.

Pentland (forthcoming) presents a detailed analogy between organizational routines and grammar, as shown in Figure 1. In that analogy, an organizational routine has a grammar. In the same way that linguists use grammar to describe certain aspects of a language, we will use grammar to describe certain aspects of a routine. What we actually observe, empirically, is never the whole routine (or a whole language), but only specific instances of it. In linguistics, these are called sentences, but in the behavioral world, we will call them *performances.* In the same way that English grammar allows speakers to produce a variety of sentences , an organizational routine allows members to produce variety of performances. Of course, a natural language like English can generate an infinite variety of sentences (due to an extremely large lexicon and a large number of rules), only some of which are ever actually uttered (Coulmas, 1981). An organizational routine would typically allow a comparatively small

number of possible performances, only some of which are enacted by the participants. Like a language, which may have several different categories of sentences (e. g., declarative, interrogative), a task unit may have several different categories of routines in its repertoire. Employees in a video store, for example, would enact different kinds of performances for sales, rentals, and returns.

Like sentences, performances may have identifiable pieces embedded within them. In linguistics, sentences are constructed from syntactic constituents such as noun phrases, verb phrases, and so on. In organization theory, it has long been observed that processes are also constructed from constituents, sometimes called "subroutines" (March & Simon, 1958). In the domain of customer service, a typical subroutine might be the validation of a credit card or some other authorization number. Finally, at the most concrete level, we have actions or moves (Pentland, 1992). Moves are like the words in the organizational lexicon from which performances are formed, and can be operationalized through a coding scheme, as described below. This multi-level analogy allows us to make the best use of the corresponding concepts from linguistics, where the unit of analysis is the sentence. In the method described here, the unit of analysis is the performance, and the goal of the analysis to induce a grammar that describes the language (organizational routine) by analyzing a set of individual sentences (performances).

(*** INSERT FIGURE 1 ABOUT HERE ***)

This analogy suggests a somewhat surprising insight: an organizational routine is not a single pattern, but rather a set of possible patterns (enabled and constrained by a variety of organizational, social, physical and cognitive structures) from which organizational members enact particular performances. There are many different variations of a customer service activity, for example, but to be recognizable as competent customer service, each instance must fall within certain bounds. Each instance is also, to a greater or lesser extent, the effortful

accomplishment of the participants and the emergent product of their interactions. Note that this definition is quite similar to that of Gersick and Hackman (1990), in that it points to similar (but not fixed) patterns that emerge through interaction. What is fixed, to some extent, is the space of possibilities for action (although that too can change, with sufficient effort). What are accomplished, with varying degrees of reflective agency and effort, are the particular patterns we observe.

A critical part of this conception of routines lies in the relationship between structure and agency. As Giddens (1984) and others have argued, rules, norms, schema, scripts, and other cognitive artifacts are *resources* for action, but they cannot be understood as *determining* action. Furthermore, individual cognitive structures of this kind are only one category of artifacts that can shape the flow of action in a situation. Pentland (1992) shows how unitary actions called moves are shaped by the physical and ritual dimensions of organizational structure, as well as by the need to draw upon the distributed cognitive resources that are distributed among individuals in the organization. In Pentland's framework, moves are constrained and enabled by features of the situation, so that a limited repertoire of moves will be available to members at any given time. The argument here is simply that the situated quality of action in organizations extends beyond individual moves to include sequences of moves. The enabling and constraining structures that typify organizational work situations (such as hierarchy, division of labor, and a myriad of task- and situation-specific particulars) naturally give rise to regular patterns of action.

Hutchins' (1991) analysis of a bridge crew manually navigating a ship when the electronic navigation system failed provides an excellent illustration. The distributed nature of the task (obtaining sightings, charting the current position, etc.) combined with the physical constraints and affordances present in the situation, created the possibility of several different ways of accomplishing their navigational task. After struggling with these possibilities for a while, the crew was able to settle on one particular division of labor and sequence of actions that

was most efficient (Hutchins, 1991). This example highlights the collective and distributed nature of many organizational routines: because no single individual could navigate the ship, the pattern of action was a collective accomplishment.

Defining routines in this way also helps to clarify and preserve the distinction between routines as sequences of action and routineness as a variable property that can be attributed to such a pattern. To reinforce this distinction, one might substitute the less value-laden term "process" for the ambiguous and somewhat pejorative term "routine." Processes can be more or less repetitive, more or less automatic, embody more or less variety, search, and so on. From this perspective, an organizational routine need not be especially "routine," as we shall demonstrate in the example that follows.

## AN EMPIRICAL EXAMPLE

Database International (DBI -- a pseudonym) creates and sells a software product for mainframe computers that we will call SystemTen. The software support group at DBI offers 24 hour-a-day technical assistance to customers who are having difficulty using their product. The staff of 25 specialists received about 100 new calls a week. Customers call the DBI software support group on a telephone "hot line" or by electronic mail, both of which were answered by one of the operators in the "customer service center". These incoming messages provided the "stimulus" necessary to invoke the customer service routine at DBI. After operators determine whether the caller has a valid service contract (which nearly all callers do), there was essentially no deliberation about how to proceed. The stimulus always elicited a similar response: the operator "opens" a call in DBI's call tracking database and assigns the problem to one of five "functional areas," where the search for a solution begins. In the sections that follow, we use data collected at DBI for two purposes: (a) to document the non-routine nature of the problem-solving work and (b) to develop a grammatical model that describes this work as a set of functionally similar patterns.

## Data

The data reported here were collected as part of a larger study of software support hot lines conducted by the first author (Pentland, 1992). The management at DBI granted a high level of access to the work unit and provided a passcard needed to gain entry and an unoccupied office in which to work. Before data collection began, all members of the DBI support group were informed that: (1) no individual was obligated to participate in any way; (2) no individual or group was being evaluated in any way; (3) confidentiality would be maintained both with other individuals and with management; and (4) the confidentiality of the organization and all persons involved would be protected with the public. The members of the DBI support group were very cooperative and generous with their time, thereby allowing the collection of the observational and archival data reported here.

Observational Data. Data collection continued for three months, five days a week, of participant observation. Given the technical nature of the work, the first author's role was predominantly that of observer, although he was sometimes asked to help with simple tasks such as copying, looking things up in manuals, or running down the hall to get help. Occasionally, he was invited to observe a situation that an informant felt might be especially interesting. Most of the time, the first author sat with a support specialist, observing what he or she did and taking detailed notes. It was usually possible to ask questions to clarify what had happened, what else might have been done, and so on. Unless the situation was urgent, informants were generally happy to discuss their work. It was also possible to retreat to an office and write up observations about situations where note-taking was not possible. At the end of each day, field notes were typed in the customary manner. Throughout the text, quotations from field notes are referenced by the site, the day, and the page number. For example, "(DBI, 6.3)" would refer to day six, page three of the field notes.

Archival Data. With the help of DBI personnel, a random sample of 335 calls was extracted from DBI's call tracking database, which contained the histories of over 20,000 problems.

Colby, Kennedy and Milanesi (1990) recommend a minimum of 50 instances as a basis for creating grammatical models of folktales, so this sample size seemed more than adequate. To extract this sample, we chose one random day per week (Monday through Friday) and selected all calls received on that day concerning DBI's main product, SystemTen. Each instance in this database is assigned a unique number and is referred to by the staff at DBI as a "call," although it would typically involve many distinct phone conversations. To get 335 calls, a total of 21 days were sampled between March 1988 and February 1989. The average elapsed time on the calls in the sample was 102 days.

The call tracking database provided convenient access to the entire history of problems that often stretched out over a period of weeks or months. The database was used by the DBI staff to keep contemporaneous notes on what steps had been taken, what needed to be done next, and so on. Observations by the first author indicated that the specialists were quite diligent about maintaining these records. Since the database served a variety of useful purposes for the support specialists and their managers, the information recorded was quite reliable, although not necessarily complete in every detail. The records cover the complete history of each problem (an essential feature for our analysis), but they limit the amount of detail that can be recovered about any particular interaction because specialists recorded only the actions and events that were needed for their own purposes. As we shall demonstrate in the analysis that follows, the tendency to omit certain actions from the database reduces the fit of our hypothesized grammar, thus understating the extent of the regularity of the actual patterns of activity.

<u>Coding the archival data</u>. The coding scheme used here was adapted from Pentland (1992) to take advantage of the call tracking database used at DBI. Pentland's (1992) analysis defined a set of basic moves that support specialists use to resolve problems, such as transferring problems to other areas, escalating to the engineering staff, and referring to other vendors. This framework was extended to include problems that were transferred to the customer for

additional information or for testing. Additional moves, such as opening the call and closing the call, were added for this analysis, as shown in Figure 2. When a support specialist at DBI changes the status of a call to reflect one of these moves (for example, escalating the call to development), their action is automatically recorded in the database in a very clear, unambiguous way. Thus, the codes could be read directly from the database records with very little need for interpretation.

(*** INSERT FIGURE 2 ABOUT HERE ***)

The archival sample was divided into two parts, a training sample that was used to develop the grammar inductively (N = 167) and a test sample that was used to test the grammar statistically (N = 168). Two coders were employed, each of which coded one half of the calls. After a brief training session, the coders achieved 96% unit-by-unit agreement (326 of 338 codes in a subset of 50 calls). The lack of agreement was due to minor unitizing errors (omitting a code from the sequence) that shifted the sequences off by one position, resulting in apparent disagreement on subsequent codes (Folger, Hewes and Poole, 1984).

**Non-routine Problem Solving Work**

As mentioned in the introduction, this task unit's work would be difficult to characterize as routine. The evidence supporting this contention is based on an analysis of the training sample (N = 167) taken from DBI's call tracking database and from observational data. This evidence is summarized in the following paragraphs.

<u>High Numbers of Exceptions</u>. There are several indicators of the frequency of exceptions in the input stream (or raw materials) to the DBI software support group. One indicator is the

frequency of duplicate calls (different customers calling with the same or similar problems). In many technical service hot lines, similar questions are common and some vendors publish newsletters of "commonly asked questions." In the training sample, there were only five calls with duplicate topics (IBM had recently released a new operating system and these customers had questions about its compatibility with DBI's software); every other topic was unique. Another indicator of task variety in software support is the frequency of new problems (calls that have not previously been encountered, for which a solution cannot be looked up or retrieved). To assess this frequency, we examined each call for evidence that the problem was recognized as one with a known solution. Seventy three of the calls (43.7%) were "new problems" that had not previously been encountered. The high frequency of new problems is partly explained by the fact that solutions for "known" problems are distributed to customers on a regular basis via maintenance tapes. Thus, customers tend to call with new or exceptional problems.

Even if a customer has a familiar kind of problem, there are frequently details about that customer's installation that make their specific problem exceptional. DBI's customers operate large data processing centers; they frequently have multiple CPUs, large telecommunications networks, and databases containing many millions of records. There are a variety of system dependent problems that result from unique combinations of processors, storage devices, networks, operating systems, and other software in use at the customer site. Even when two customers have a similar kind of problem, the similarity can be difficult to detect. Our assessment of the frequency of new problems tends to understate the variety introduced by these contingencies.

Low Task Analyzability. Although the world of computer software can sometimes be highly rationalized and well-understood, the task environment at DBI had certain features that made many problems difficult to analyze. First, the product was very complex, as this engineer explained:

> I can tell you that SystemTen is off the scale in terms of complexity. There's no way on earth to create enough test cases for it. There's no doubt that it's beyond the realm of human understanding. Certain modules you just have to put a skull and crossbones on and treat as a black box. (DBI, 34.6)

As a result, it was not always possible to be sure of what caused a problem, what a feasible solution might be, or even where to look. This was especially true of "data dependent" or timing errors, which could usually only be reproduced on the customer site. Second, it was sometimes difficult to construct an explanation for a problem even when the symptoms were known, as this engineer explained:

> Even if you have the dump, all you really have is a footprint in the sand. You can see that it happened, but you have no idea why it happened. (DBI, 5.13)

The "footprints in the sand" phenomenon can be detected in the training sample taken from DBI's call tracking database. Out of these 167 calls, forty seven (28.1%) were never solved. Either the problem was a "one time occurrence" for which no meaningful diagnostic data could be collected or the customer upgraded some aspect of their system and the problem went away. Finally, it was not always possible to obtain enough information from users to isolate the symptoms of a problem, much less to explore its causes. In these cases, there was no way to analyze what had happened.

Extensive Deliberation and Search. Given the complexity of the problems being analyzed, it should not be surprising that considerable deliberation and search was required. The first author often observed technical support specialists working several hours on a single problem, sifting through "dumps" and "snaps" (foot-thick stacks of paper filled with hexadecimal representations of the file buffers, main memory, and other diagnostic information), or stepping through programs one instruction at a time. Other members of the support staff were frequently consulted, and calls were frequently transferred between functional areas or

escalated to the developers who wrote the code so that the appropriate expertise could be brought to bear on the problems. Because so many problems were site-specific, it was frequently necessary to transfer responsibility for a problem to the customer who reported it, so that more information could be gathered. Customers were sometimes asked to recreate a problem or to run a test procedure and report the results, for example.

In no sense could this work be described as mindless or automatic. When the source of a problem was identified, there was an additional phase of deliberation involved in deciding what to do about it. In one particularly difficult case, several meetings were required to decide if the fix should be issued for the current version of the product (as a "patch") or delayed until the new version. In some cases, a customer could simply "work around" the problem. In other cases, a problem could be fixed by supplying a customer with a customized "ZAP" that changed their copy of the program. In still other cases, it might require more extensive design changes to the underlying product. Choosing among these alternatives often required that a problem be escalated to higher level technical or managerial personnel. Like Hutchins' (1991) navigational task, this work required the collaboration of many individuals; no individual could solve all of the problems.

Taken together, these indicators strongly suggest that this task unit's work was non-routine. In terms of definitions based on Perrow (1967), this unit had high task variety and moderate analyzability of search, at best. They operated with considerable deliberation and search and frequent interruptions, thus violating Ashforth and Fried's (1988) criteria for routinized behavior. Given these findings, we might be tempted to speculate that this task unit had no routine, no functionally similar pattern that characterized it work processes. This conclusion, however, would be based on an unwarranted inference from the properties of the work to the sequential structure of the actions required to accomplish it. In the following section, we develop a grammatical model and use it to test for repetitive, sequential patterns of action in the work.

**Constructing the Grammatical Model**

There are several steps required to construct a grammatical model of an organizational process. They are described and illustrated here in sufficient detail to allow replication. Some of these steps will be familiar, such as the basic coding procedure described above. In this respect, this method builds on a variety of familiar techniques for handling sequential data (Gottman and Roy, 1990; Van de Ven and Poole, 1990). The more specialized steps involved in creating a grammar and fitting it to the data will be less familiar and will receive special attention here.

Exploring the data. As with any inductive procedure, the first step is to become familiar with the data. The first twenty coded calls from the training sample are shown in Figure 3. Each row represents a single call, which is one performance of the software support routine. The symbols in each row represent moves (or actions) that were taken in the process of responding to the call. In the discussion that follows, the coded data will be referred to as "strings" or "sequences." The sequence in row (1) can be read as follows: the call was opened (O), worked on two different times (W W), transferred to the user (US), transferred back to product support (PS), worked on two more times (W W), and then closed (C). In general, the coding procedure reduces a great deal of other information down to these strings of symbols. For example, we record the sequence of events but not the interval between them (Abbott, 1991, refers to this as "event time" as opposed to "clock time") and a wide variety of work activities is compressed into a single category (W).

(*** INSERT FIGURE 3 ABOUT HERE ***)

Nonetheless, the variety among the performances in Figure 3 is striking. They are all constructed from the same basic lexicon of moves, but the sequences vary considerably from

performance to performance. Of these 20 calls, only the five simplest calls had the same pattern (lines 4, 8, 11, 12 and 18) (none of these was the same problem, by the way). All of the other calls entail unique sequences of action, yet they all accomplish the "functionally similar" objective of responding to a customer problem. Superficially, these data seem to support the conclusion that this task unit has no routine; there seems to be too much variance in the data. The goal of the next step in the analysis is to summarize this sequential variance using a simple grammar.

Create a Grammar that Describes the Pattern. As we explore the data, we will be working towards a formulating a hypothetical grammar. The grammar will consist of a set of phrase structure rules (sometimes called "rewrite rules," for reasons that will become apparent in a moment) that specify patterns in the coded data. Note that these rules need not be thought of in the usual sense of rules that organizational members follow or refer to while doing their work. Members of the DBI technical support group do not follow these rules, any more than they would follow the rules implicit in a conventional regression model. Rather, phrase structure rules of the kind used here are simply devices for describing patterns observed in the coded data.

Formulating a grammar is something of an art. One can begin to formulate a grammar from the top down, by starting with the whole routine and breaking it into constituent parts. Alternatively, one can work from the bottom up, by starting with the coded data and identifying common sub-sequences. A top-down strategy would be most appropriate when the overall logic of the routine is known in advance, or when one has a priori expectations based on theory. A bottom-up strategy would be more appropriate when one has no a priori theory and wants to work inductively from the data. In either case, the goal is to account for the largest number of observations with the simplest set of rules. These rules form a kind of working hypothesis about the sequence of actions observed in the routines.

Whether one is working top-down or bottom-up, it may be helpful to look for the constituent parts of the routine. These are like the "syntactic constituents" (Cook, 1988) that make up a

formal grammar. These are sub-patterns that can be meaningfully isolated from the rest of the sequence. For example, each software support call consists of an opening, followed by a search for the solution, followed by a closing. In grammatical form, this could be expressed as:

**Software Support Call --> Opening, Search for Solution, Closing.**

where the symbol "-->" can be read as "consists of" and the commas in the list indicate "followed by." This is a simplified version of how phrase structure grammars (Gazdar et al., 1985) are used to represent sentences. Each rule has a single symbol on the left-hand side, and one or more symbols on the right-hand side. In English grammar, for example, one might have a rule that says that a sentence (S) can consist of a noun phrase (NP) and a verb phrase (VP). Such as rule could be written as follows: S --> NP, VP. One can think of these phrase structure rules as describing how a particular entity (such as a customer service interaction) can be expanded into its constituent parts.

The next step in developing a grammar is to enumerate the ways in which each constituent can be accomplished. In this respect, the method shares some common ground with ethnographic procedures for mapping means-ends semantic domains (Spradley, 1979). The semantic domain of interest here can be thought of as the "ways to accomplish" the constituent in question. In general, there may be several possible ways to accomplish a given constituent of a routine. To model this, one simply includes alternative expansions of that constituent; each possible alternative gets one rule. Theoretically, these alternatives are functionally similar in the sense suggested by Gersick & Hackman (1990) because they satisfy a common means-ends relationship. Every retail transaction, for example, involves form of payment, but there are usually several ways that payment can be accomplished (e.g., cash, check, credit card, etc.). In Spradley's (1979) terms, these alternatives comprise the domain, "ways of paying." In our terms, they would be included as separate rules in a grammar of retail transactions (Pentland, forthcoming). Each constituent must be expanded in this manner until it can be expressed in

terms of the basic lexicon or coding scheme. This is what allows the grammar to be tested against the data, as described below.

This procedure was used to induce a hypothetical grammar for the DBI software support process, as shown in Figure 4. The particular rules we have identified were developed by carefully examining the sequences in the training sample and applying insights gained from the observational data. As the hypothetical grammar took shape, we applied it to the training sample to see how well it fit the data. Like any inductive exercise, a certain amount of judgment was required. The first rule in Figure 4 states that the overall structure of the pattern consists of an opening, followed by a search for the solution, followed by a closing. Since the opening was coded with a single symbol in the lexicon, the rule that expresses the opening is trivial: opening --> O. If more detailed data were available for each call, this constituent could have been coded in more detail to include the way calls are answered, the initial conversation with the caller, and the steps required to enter a new call into the tracking database. In that case, we would add rules to express the alternative ways that an opening could be accomplished.

The body of each call consists of a sequence of moves that represent the process of searching for a solution. The search process can include any sequence of transferring, escalating, or working on the problem. In addition, there are some special constituents within the search process, that we have labeled "user" and "developer." For example, in the second line of Figure 3 the sequence "US PS" occurs. This means that responsibility for the problem was turned over to the user (US) and then taken back by product support (PS). This sequence happens frequently in problems where the only way to advance the solution is to have the customer gather more information or test something on their system. When the information arrives or the test is completed, the specialist assumes responsibility again. Figure 4 also includes a similar set of patterns between the developer (DV) and product support (PS) that we have labeled "developer" constituents.

(*** INSERT FIGURE 4 ABOUT HERE ***)

Each call ends with a closing sequence that involves a tentative fix, followed by an action indicating that the call is "done", followed by a trailing comment that explains what was done. Observations of the work process and an examination of the call data indicated that closing a call does not necessarily imply solving the problem. Many of the calls that are handled by DBI are reports of one-time problems that are never duplicated and never solved. Nonetheless, these calls are usually either closed or become inactive. For this reason, the grammar includes an alternative sequence that omits the tentative fix. Like the user and developer episodes, "tentative fix" and "done" are intermediate level constituents that have an internal structure that is expressed by further rules, as shown in Figure 4. A tentative fix contains one of three codes indicating that a fix has been tried (F or R or T) followed by a work episode (usually involving a test of the fix). The "done" constituent is represented by either C (closed) or D (deferred) or I (inactive). In some calls, there is another symbol indicating an additional comment that explains what steps were taken to close the call (E).

Rewriting. The procedure used to test the grammar against the data is called "rewriting", since it consists of taking each coded string and literally rewriting it using the hypothesized rules. Rewriting is very similar to what we would normally describe as coding. One searches data for a specified pattern, but instead of simply annotating the data, one actually replaces the original data with the code (hence the term rewriting). For simple rules, one could accomplish this with the "search and replace" option available in any word processing program. In rewriting, one looks for patterns in the data that match the right-hand side of a rule. When a match is found, that part of the data is replaced with the left-hand side of the rule (the name of the constituent). In this way, detailed patterns in the data are replaced by the more abstract constituents. This basic procedure is repeated for each of the rules in the grammar until no

further rewriting is possible. This can be accomplished manually, or it can be accomplished using a computer program designed for pattern matching and re-writing.[1]

Figure 5 illustrates the procedure for rewriting a single string, one rule at a time. The first row in Figure 5 shows the raw data, consisting of a sequence of seven moves. Each successive row shows the same data after it has been rewritten by a single rule. The vertical lines trace the path of each data element as it is transformed by the rewriting process. Double lines indicate that an element has been rewritten. If this figure were inverted, it would resemble a simple sentence diagram. Thus, in the second row, note that the "O" has been rewritten using the rule "Opening --> O." In the next row, the "C" has been rewritten using the rule "Done --> C." The ability of rewriting to summarize patterns is demonstrated in the fourth row, where the rule "Closing --> Done, Comment" was used to replace two constituents with one, as indicated by the converging lines. In each of the next three rows, two symbols were re-written to a single symbol, as indicated by the converging lines. In the final row, three constituents were unified into a single symbol that represents the overall pattern.

If a string is completely rewritten to a single symbol, as in this example, then it fits the grammar perfectly and can be said to have been "parsed" by the grammar. This would be like having a data point in a regression that fits the model with an error term of zero. The number of strings in a sample that are completely rewritten by the grammar is one measure of how well the grammar fits the data; we will use this measure in the analysis that follows. Unlike this example, however, not all observed sequences can always be completely rewritten. There may be sequences in the data that do not fit the hypothetical grammar. Since rewriting tends to

---

[1]For the analysis that follows, we used a program written in MacPROLOG 4.0 (Quintus Corp.), and run on a Macintosh IIci personal computer.

reduce the length of the string being rewritten, the percentage reduction in length is another indicator of the degree of fit (Olson et al, forthcoming).

(*** INSERT FIGURE 5 ABOUT HERE ***)

Testing the grammar. Our hypothesis is that in spite of frequent interruptions and extensive search, the support process embodied sequential patterns that could be represented by a grammar. A convenient null hypothesis that allows us to formulate a statistical test is that the observed sequences of action are random and do not follow the hypothesized pattern. To test whether the service process at DBI conformed to our hypothesized grammar, we performed the rewriting procedure on second half of our coded data, the test sample (N = 168). Note that these data were not included in the training sample that was used to formulate the hypothetical model. One hundred nine of the strings, or about 64.9%, were rewritten completely. In the remaining strings, most were rewritten to some extent, but not always with much reduction in the length of the string of symbols. These ungrammatical cases are discussed below.

To test the statistical significance of this result, we needed to estimate the probability that it could be obtained by applying the grammar to a set of random sequences. Since no probability distributions are available for this kind of data, we performed a simple bootstrap procedure of the kind described by Olson et al (forthcoming). We generated 100 sets of random data by creating random permutations of each string in the real dataset. Thus, each randomized dataset consisted of 168 strings, each with the same length and composition of elements as the original. (To visualize this procedure, imagine taking the sentences in this paragraph and randomly rearranging the words in each sentence 100 times to create 100 new paragraphs.) We applied the hypothesized grammar to each of the 100 randomized datasets; the mean number of strings

completely rewritten was 9.57 (out of 168), with a standard deviation of 2.77. The observed result from the real data (109 out of 168) was over 35 standard deviations higher than the mean.

Analysis of ungrammatical patterns. The question remains, of course, whether some other grammar might be better or more parsimonious than our hypothesized grammar. To help answer this question, it is useful to analyze the residual patterns that do not fit the hypothesized grammar. The most common disparity in the test sample (appearing in twenty four of the incompletely rewritten calls) was the presence of a single "US" without any corresponding "PS." This pattern would imply that responsibility for the call was transferred to the user and never taken back by DBI's product support group. In nearly all of these instances, a DBI specialist had implicitly taken responsibility by continuing to work on the call or taking some other action, but had neglected to indicate this in the database. The existence of this pattern makes sense, because the high workload encourages support people to give problems away and to take them back only if necessary. This omission on the part of the DBI staff significantly reduces the apparent fit of the grammar. If the data had reflected this move, then the percentage of calls completely rewritten would exceed 79%.

The remaining few calls reflected more fundamental departures from the grammar. For example, seven of the calls had been closed and then reopened. In these cases, problems that seemed to have been solved turned out to require additional work. Thirteen of the calls had never been officially closed (or deferred or declared inactive), although work had apparently ceased. Many of these calls had been transferred to the customer for further data which apparently never materialized. Without customer pressure on these problems, the support people simply attended to more pressing matters.

These exceptional patterns are theoretically important because they indicate that even the simple structure that we have hypothesized is not always attained in practice. They also suggest ways in which the hypothesized grammar might be improved. One might want to add a rule, for

example, that allows for user episodes that do not include returning responsibility to product support. One might also want to add a rule that allows patterns that have no closing constituent, or patterns that are closed and then re-opened. These rules would improve the coverage of the grammar to over 90% of the test sample, but they would also contradict our normative sense, based on insights from the fieldwork, of what constitutes a complete, competent call. Furthermore, there is always the danger of overfitting the model; in principle, one can simply keep adding rules until every possibility is covered. The decision of which rules to include and which to exclude is essentially a judgment call. Although we can test a hypothetical grammar against a null hypothesis, we do not yet have tests to compare the relative performance of different grammars (Simon, 1992).

## DISCUSSION

This grammatical analysis reveals a great deal of regularity in data that would appear superficially to be quite unstructured. Even though the software support performances (as shown in figure 3) appear to be quite diverse, they share an underlying structure that can be captured in the rules of a grammar. We are forced to conclude that although this work is apparently non-routine, it embodies a repetitive, functionally similar pattern of action.

The observed regularity arises from the way the work is organized (e. g., division of labor), structural features of the situation and the cognitive models of the participants. Calls must be transferred between specialists, information must be solicited from the user, calls must be opened before they can be closed, and so on. If our dataset allowed us to code the process at a finer level of detail, additional regularities would emerge. It could hardly be otherwise, because of the way in which routines are embedded in organizations. As Winter (1986: 168) notes, routines are often associated with specialized human or physical capital that he calls "routine-specific assets." These assets define a set of possible actions which give rise, in practice, to repetitive patterns.

In the DBI software support group, for example, specialists have a limited lexicon of moves with which to accomplish their work. At any point in time, there are only a few choices about how to proceed with solving a problem, as reflected in the lexicon. There may be a great deal of creativity and diligence applied toward this goal, but the generic choices about what can be done next are quite limited. Further, there is a logical dependence among some of these moves. For example, once a call is transferred to a user, it must be transferred back. These dependencies give additional structure to the performances, but they fall far short of constraining all performances to be identical. What we find, instead, can best be described as *constrained variety;* performances that are functionally similar, but not necessarily the same. Finally, these patterns will emerge whether or not the preconditions for purely scripted behavior are present. They arise from the way the work is organized, as reflected in the constraints and affordances of the situation. This interpretation is quite consistent with Nelson and Winter's (1982) ideas about routine-specific assets, routines as truce, and other contextual features that enable and constrain action. Cognitive structures are also important, of course, insofar as they allow the agents in the situation to employ the appropriate assets, observe the truce, and thereby produce the desired pattern of action.

Conceptualizing routines grammatically draws attention to the relationship between an organization's structure and the sequences of action that occur there. In this framework, structure defines the set of possibilities, but not the particular sequences we observe. Thus, it becomes natural to ask how changes in structural features might alter the observed pattern of action. For example, might a new way of communicating and sharing information with users and developers alter the pattern of work? How would a different division of labor affect the patterns of work? How persistent are routines in the face of changing institutional structures? To even begin to address such questions, one must be able to summarize the patterns of action that make up the hypothesized routines.

Grammatical models have distinct advantages over other kinds of models (e.g., Markov models) for this purpose. First, they describe a set of possible patterns of action from which members enact particular patterns. This formulation has theoretical value, because it accommodates the apparently conflicting perspectives on routines as fixed patterns and routines as effortful accomplishments. It has methodological value, as well, because real routines are unlikely to display "fixed" patterns. Rather, they display a variety of functionally similar patterns that can be concisely summarized using the kind of formalisms described here. Second, phrase structure rules provide a convenient way to describe the layers of constituents ("sub-routines") that can be rearranged and combined to form new patterns. Each constituent can be expanded into a family of functionally similar patterns that represent alternatives ways of accomplishing that part of the routine. Third, as illustrated in the case of the user and developer episodes, grammatical models are well suited to representing connections between events that may be widely separated in a sequence. Fourth, grammatical models provide a clear way to distinguish normatively correct instances of a routine from other instances, as illustrated in the tendency of the DBI support staff to avoid claiming responsibility for certain calls.

Last but not least, the use of a grammatical model allows us to operationalize the definition of an organizational routine as a set of *repetitive, functionally similar patterns of action* with a degree of precision that has not been possible before. In particular, a sequence of coded symbols is a "pattern of action" if it can be parsed by a grammar. The patterns of action within the grammar are established as "functionally similar" by their inclusion within a common means-ends semantic domain (Spradley, 1979). That is, each pattern in the grammar is a "means" to a particular "end." And finally, a pattern of action is "repetitive" if one can collect multiple instances of it within some time period of practical or theoretical interest. While still subject to interpretation and refinement in particular settings, these operationalizations help to make the concept of organizational routines more explicit by defining empirical criteria for what counts as a routine and what does not.

**A New Comparative Framework**

If we carry the analogy to language forward, it suggests the possibility of new kind of comparative framework for organizational analysis. In human language, there are grammatical similarities between different languages, although the actual words are completely different. By analogy, the customer service process in many organizations may have structural similarities, although the specific actions taken in each organization may be quite different. The same may be true of other kinds of processes, as well, such as sales, purchasing, recruiting, auditing, planning, and so on. A potentially powerful advantage of a grammatical approach is the ability to summarize innumerable diverse patterns of activity into concise, high-level patterns. In the same way grammar allows linguists to identify families of human languages that share certain syntactic characteristics (Newmeyer, 1983), a grammar of routines should allow organizational scholars to identify families of organizational processes and to chart their changes over time. The grammatical framework becomes especially valuable when there is a profusion of lexical items, such as nouns. By creating categories of functional equivalence, grammar facilitates comparisons across sets of apparently diverse instances.

As a practical matter, grammatical models seem most appropriate for situations where sequential records are kept or can be easily generated. Observational data is expensive, especially for processes that are distributed in time and space. Fortunately, many kinds of business processes generate archival data as a byproduct of normal operations which could be analyzed using these methods. Bill collectors, for example, keep records of their activities that could potentially be useful (Sutton, 1991). With the increased use of information systems in all kinds of service and manufacturing organizations, the necessary data is likely to become more and more available.

Assuming that data are available, the success of this approach will depend on our ability to formulate an appropriate lexicon and syntactic constituents. These intermediate levels of abstraction are analogous to the constructs one finds in traditional nomological research, except

that instead of referring to the variable properties of objects, they refer to actions or collections of actions. By introducing abstract constituents (e.g., “opening”, “search” and “closing”) to represent parts of a routine, it allows routines that differ in their details to be compared in terms of their constituents. In linguistics, it is the existence of syntactic constituents such as noun phrases and verb phrases that forms the basis for all subsequent theorizing (Cook, 1988). It is the arrangement of these constituents that characterizes the structure of a language, much more so than the details of the vocabulary or other aspects of surface structure. In the case of software support, the simple three-part structure for calls (opening, search for solution, closing) suggests the possibility of comparing the ways that different organizations accomplish openings, searches, or closings, independent of how the other parts of the overall routine are accomplished.

**Assessing Alternative Models**

Although organizational routines might be expected to have a great deal of regularity, there is always the possibility of exception, breakdown, creativity, shirking, error, and other variety-producing contingencies. It seems unreasonable to expect that all instances of a routine would fit an hypothesized model. It would be highly desirable to have a systematic way of ascertaining whether an hypothesized model could be improved upon by adding or deleting rules. Simon (1992) notes that the best available test for the adequacy of rule based models is the general heuristic that the model should explain many more cases than the number of rules. Using this rule of thumb, our hypothesized model fares reasonably well by explaining 109 cases (plus a similar number in the training sample) with only 13 rules. There is clearly a trade-off here between the complexity of the grammar and the number of cases it can describe, but we currently have no statistical test for assessing the relative contribution of individual rules.

## Some Issues and Assumptions

The key assumption of this method is that organizational life can be coded into meaningful syntactic constituents. This assumption is no different, in principle, than the assumption that survey responses mean the same thing to different respondents and can therefore be compared and aggregated. While this assumption has been criticized (Cicourel, 1964; Blumer, 1968), it is more commonly treated as completely unproblematic. As in any methodology that involves a representation of a complex phenomenon, the representation tends to distance us from the object of our inquiry. The advantage of this approach is that unlike traditional linear models, which at best employ summary measures of the overall routine, the essential sequential structure of the routine is preserved. The disadvantage is that the experiences of the participants are reduced to strings of coded symbols.

This method also assumes that it is meaningful to isolate the synchronic structure of routines from their diachronic context (Barley, 1990). As Fabian (1990) notes, the performances that we observe are always just the tip of an iceberg that consists of the rehearsal and repetition of these patterns over time; these performances create the context in which they occur just as much as they are created by it. We would argue, however, that it is helpful to bracket off the diachronic in order to achieve a degree of clarity about the patterns of action we observe at any particular moment in time. This framework can be used to describe lexical evolution (new moves) and syntactic evolution (new patterns of existing moves), but the source of these changes would be exogenous to the model.

Finally, there is always the risk that describing patterns of social action with grammar will be misinterpreted as an attempt to objectify social life in a way that is inconsistent with our best understanding of the nature of the social world. Rules are resources for action, but they do not determine action (Zimmerman, 1970; Giddens, 1984). Nonetheless, there is a sense in which rules can embody physical or logical constraints on the possibilities for action in a situation. An example of such a rule (from Salancik & Leblebici, 1988), is that one cannot eat a meal

before it has been prepared. Note that this rule does not force one to eat a meal just because it has been prepared, it simply rules out the possibility of reversing this sequence. It is not deterministic, but it does constrain the set of possibilities. In recognition of these issues, our methodology makes no attempt to uncover any rules that people "really use." Although we use rules to describe patterns of action, this is merely a convenient way of summarizing sequential patterns of action that allows comparison and generalization across settings.

For this reason, the models presented here are rather different in spirit than many of the symbolic or computational models that have been used in cognitive science and organization theory. Unlike Simon (1969, 1992), we are not attempting to create a reductionist model of individual cognitive processes that, when confronted with a set of situational constraints, reproduces a particular pattern of behavior. Simon (1969) argues that the circuitous path of an ant carrying a crumb, for example, is the product of the ant's simple rules for navigation applied to the constraints of the environment. While people have more extensive cognitive capabilities than ants, members of organizations do work in constrained environments that limit their possibilities for action. Grammatical models of organizational routines provide a convenient way to summarize those possibilities and render them available for inspection and analysis, but they leave the enactment of particular patterns open to unfold anew each time. In this respect, grammatical models capture the essence of the phenomenon, as eloquently expressed by one of the anonymous ASQ reviewers: "Routines are like ruts in a well-traveled road. They do not exactly determine where the next wagon will go, but neither do they merely describe where past wagons have gone."

## CONCLUSION

Imagine a community of scholars for whom color is a critical concern. These scholars place color at the center of some of their most important theories, yet they have no words for red, yellow, green or blue and no way of distinguishing among them. Organizational scholars are not

quite this bad off, of course, but our tools for describing and generalizing about routines are weak, at best. Our theories refer to routines as though they were concrete objects, yet we have no systematic means of describing these objects. By providing a way to represent the sequential structure of routines, grammatical models create the possibility of investigating core issues in organization theory in ways that were not previously possible. It remains to be seen whether the patterns of action we call routines merit the theoretical importance we have given them, but without a methodological lens to help us see these patterns, we might never know.

## REFERENCES

Abbott, Andrew

1991 "A primer on sequence methods." Organization Science, 1: 375-392.

Alker, Hayward R., Jr.

1987 "Fairy Tales, Tragedies, and World Histories: Towards Interpretive Story Grammars as Possibilist World Models." Behaviormetrika, 21: 1-28.

Ashforth, Blake E. and Yitzhak Fried

1988 "The Mindlessness of Organizational Behaviors." Human Relations, 41: 305-329.

Barley, Stephen R.

1990 "Images of Imaging: Notes on Doing Longitudinal Fieldwork." Organization Science, 1: 220-247.

Blumer, Herbert

1969 Symbolic Interactionism: Perspective and Methods, Englewood Cliffs, NJ: Prentice-Hall.

Chomsky, Noam

1956 "Three models for the description of language." IRE Transactions on Information Theory, IT-2: 113-124.

Cicourel, Aaron

1964 Method and Measurement in Sociology, New York: Free Press.

Cohen, Michael D.

1991 "Individual Learning and Organizational Routine: Emerging Connections." Organization Science, 2: 135-139.

Colby, Benjamin N., Sayuri Kennedy and Louis Milanesi

1991 "Content Analysis, Cultural Grammars, and Computers." Qualitative Sociology, 14: 373-387.

Cook, V. J.

1988 Chomsky's Universal Grammar, Oxford, England: Basil Blackwell.

Coulmas, Florian
1981 Conversational Routine: Explorations in Standardized Communication Situations and Prepatterned Speech, The Hague: Mouton.

Daft, Richard L. and Norman B. Macintosh
1981 "A Tentative Exploration into the Amount and Equivocality of Information Processing in Organizational Work Units." Administrative Science Quarterly, 26: 207-224.

Fabian, Johannes
1990 Power and Performance: Ethnographic Explorations through Proverbial Wisdom and Theatre in Shaba, Zaire, Madison, WI: University of Wisconsin Press.

Folger, Joseph P., Dean E. Hewes and Marshall Scott Poole
1984 "Coding Social Interaction." In Brenda Dervin and Melvin J. Voight, (eds.) Progress in Communication Science, Volume IV: 115-161. Norwood, NJ: Ablex.

Garfinkel, Harold
1967 Studies in Ethnomethodology, Englewood Cliffs, NJ: Prentice-Hall.

Gazdar, Gerald, Ewan Klein, Geoffrey Pullum and Ivan Sag
1985 Generalized Phrase Structure Grammar, Cambridge, MA: Harvard University Press.

Gersick, Connie J. and J. Richard Hackman
1990 "Habitual Routines in Task-Performing Groups." Organizational Behavior and Human Decision Processes, 47: 65-97.

Giddens, Anthony
1984 The Constitution of Society, Berkeley, CA: University of California Press.

Gioia, D. and P. Poole
1984 "Scripts in Organizational Behavior," Academy Management Review, 9, 449-459.

Goffman, Erving
1981 Forms of Talk, Philadelphia: University of Pennsylvania Press.

Gonzalez, Rafael C. and Michael G. Thomason
1978 Syntactic Pattern Recognition: An Introduction, Reading, MA: Addison-Wesley.

Gottman, John M. and Anup K. Roy

1990 Sequential Analysis: A Guide for Behavioral Researchers, Cambridge: Cambridge University Press.

Hannan, Michael T. and John R. Freeman

1983 "Structural inertia and organizational change." American Sociological Review, 29: 149-164.

Heritage, John

1984 Garfinkel and Ethnomethodology, Cambridge, England: Polity Press.

Hutchins, Edwin

1991 "Organizing Work by Adaptation." Organization Science, 2: 14-39.

Leidner, Robin

1993 Fast Food, Fast Talk: Service Work and the Routinization of Everyday Life, Berkeley: University of California Press.

Leigh, Thomas W. and Patrick F. McGraw

1989 "Mapping the procedural knowledge of industrial sales personnel: A script-theoretic investigation." Journal of Marketing, 53: 16-34.

Levitt, Barbara and James G. March

1988 "Organizational Learning." Annual Review of Sociology: 319-340.

Lynch, Beverly P.

1974 "An Empirical Assessment of Perrow's Technology Construct." Administrative Science Quarterly, 19: 338-356.

March, James G and Herbert A. Simon

1958 Organizations, New York: John Wiley and Sons.

Miclet, Laurent

1986 Structural Methods in Pattern Recognition, New York: Springer-Verlag.

Nelson, Sidney G. and Richard R. Winter

1982 An Evolutionary Theory of Economic Change, Cambridge, MA: Harvard University Press.

Newmeyer, Frederick J.

1983 Grammatical Theory: Its Limits and Its Possibilities, Chicago: University of Chicago Press.

Olson, Gary M., James D. Herbsleb and Henry H. Rueter
Forthcoming "Characterizing the Sequential Structure of Interactive Behaviors through Statistical and Grammatical Techniques." Human Computer Interaction.

Pentland, Brian T.
1992 "Organizing Moves in Software Support Hot Lines." Administrative Science Quarterly, 37: 527-548.

Pentland, Brian T.
Forthcoming "Grammatical Models of Organizational Processes." Organization Science.

Perrow, Charles
1967 "A Framework for the Comparative Analysis of Organizations." American Sociological Review, 32: 194-208.

Prince, Gerald
1973 A Grammar of Stories, The Hague: Mouton.

Ryan, Marie-Laure
1979 "Linguistic models in Narratology: From structuralism to Generative Semantics." Semiotica, 28: 127-155.

Salancik, Gerald R. and Husseyin Leblebici
1988 "Variety and Form in Organizing Transactions: A Generative Grammar of Organization." In Nancy DiTomaso and Samuel B. Bacharach, (eds.) Research in the Sociology of Organizations: 1-32. Greenwich, CT: JAI Press.

Sandelands, Lloyd E.
1987 "Task Grammar and Attitude." Motivation and Emotion, 11: 121-143.

Schank, Roger C. and Robert P. Abelson
1977 Scripts, Plans, Goals and Understanding: an Inquiry into Human Knowledge Structures, Hillsdale, NJ: Lawrence Erlbaum.

Schegloff, Emanuel A.

1982 "Discourse as an interactional achievement: Some uses of 'uh-huh' and other things that come between sentences." In Deborah Tannen, (eds.) Georgetown University Round Table on Languages and Linguistics 1981: 71-93. Washington, DC: Gèorgetown University Press.

Simon, Herbert A.
1957 Administrative Behavior, New York: Macmillan.

Simon, Herbert A.

1969 The sciences of the artificial, Cambridge, MA: MIT Press.

Simon, Herbert A.
1992 "What is an "explanation" of behavior?" Psychological Science, 3: 150-161.

Spradley, James P.
1979 The Ethnographic Interview, New York: Holt, Rinehart, Winston.

Sutton, Robert I.
1991 "Maintaining norms about expressed emotions - The case of bill collectors." Administrative Science Quarterly, 36: 245-268.

Van de Ven, Andrew H. and Marshall Scott Poole
1990 "Methods for Studying Innovation Development in the Minnesota Innovations Research Program." Organization Science, 1: 313-335.

Ventola, Eija
1987 The Structure of Social Interaction: A Systematic Approach to the Semiotics of Service Encounters, London: Frances Pinter.

Walsh, James P. and Gerardo R. Ungson
1991 "Organizational Memory." Academy of Management Review,16: 57-91.

Weick, Karl E.
1979 The Social Psychology of Organizing, Second Edition, New York: Random House.

Weiss, Howard M. and Daniel R. Ilgen
1985 "Routinized behavior in organizations." Journal of Behavioral Economics, 14: 57-67.

Winter, Sidney G.

1986 "The research program of the behavioral theory of the firm: Orthodox critique and evolutionary perspective." In B. Gilad and S. Kaish, (eds.) Handbook of Behavioral Economics, Volume A: Behavioral Microeconomics: 151-188. Greenwich, CT: JAI Press.

Withey, Michael, Richard L. Daft and William H. Cooper

1983 "Measures of Perrow's Work Unit Technology: An Empirical Assessment and a New Scale." Academy of Management Journal, 26: 45-63.

Zimmerman, Donald H.

1970 "The Practicalities of Rule Use." In Jack Douglas, (ed.) Understanding Everyday Life: Toward the Reconstruction of Sociological Knowledge: 221-238. Chicago: Aldine.

**Figure 1: Analogy between behaviorial and grammatical entities**

| Behavioral Entity | Grammatical Counterpart |
|---|---|
| **Organizational Routine**: the set of possible performances for a particular task, described in part by a grammar.<br><br>Example: customer service | **Language**: the set of possible sentences in a particular language, described in part by a grammar.<br><br>Example: English |
| **Performance**: a single, complete repetition of a routine.<br><br>Example: a particular customer service encounter. | **Sentence**: a single, complete expression in a language.<br><br>Example: "The dog is barking." |
| **Subroutine**: intermediate part of a routine that acts as a building block.<br><br>Examples: Answering the phone, authorizing a credit card. | **Syntactic consitutent**: intermediate part of a sentence that acts as a building block.<br><br>Examples: noun phrase ("the dog"), verb phrase ("is barking"). |
| **Move**: unitary act that comprises the routine.<br><br>Examples: Greet customer, transfer customer. | **Word**: Unitary element that comprises a language.<br><br>Examples: "dog", "barking." |

**Figure 2: Coding scheme for software support prccess**

| Code | Explanation |
|---|---|
| O | Open the call |
| W | Work on the call |
| C | Close the call |
| I | Declare problem inactive (usually because it can't be reproduced, so no further work can be done) |
| D | Defer the problem; can't solve now |
| T | Tentative fix completed |
| R | Resolve problem (but leave call open for confirmation by customer) |
| F | Fix given to customer |
| E | Explain reason for closing |
| US | Transfer responsibility to user |
| PS | Transfer responsibilty to product support |
| DV | Transfer responsibility to development |
| TR | Transfer call to another functional area |

**Figure 3: Sequence of moves in subset of the call sample**

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1) | O | W | W | US | PS | W | W | W | C | | | |
| 2) | O | W | W | C | E | | | | | | | |
| 3) | O | W | W | W | W | W | W | W | W | C | E | |
| 4) | O | W | C | | | | | | | | | |
| 5) | O | W | C | E | | | | | | | | |
| 6) | O | W | W | W | C | | | | | | | |
| 7) | O | W | W | TR | W | C | | | | | | |
| 8) | O | W | C | | | | | | | | | |
| 9) | O | W | US | PS | W | W | W | TR | W | US | W | C |
| 10) | O | W | W | US | PS | W | W | DV | US | W | W | C |
| 11) | O | W | C | | | | | | | | | |
| 12) | O | W | C | | | | | | | | | |
| 13) | O | W | W | US | PS | C | | | | | | |
| 14) | O | W | W | W | C | | | | | | | |
| 15) | O | W | TR | W | W | W | C | | | | | |
| 16) | O | US | PS | W | W | I | | | | | | |
| 17) | O | W | TR | W | I | | | | | | | |
| 18) | O | W | C | | | | | | | | | |
| 19) | O | TR | W | W | W | US | W | PS | W | TR | W | C |
| 20) | O | W | US | W | W | C | | | | | | |

**Figure 4: A grammar for DBI's software support process**

| Grammatical Rule | Explanation |
|---|---|
| CALL --> opening, search for solution, closing. | A whole call, consisting of a opening, search for solution and closing. |
| opening --> O. | Opening move of a call. |
| search for solution--> any order( work, transfer, user, developer ). | Search for solution, consisting of any sequence of working, transferring, as well as user- and developer- episodes. |
| closing --> tentative fix, done, comment.<br>closing --> done, comment.<br>tentative fix --> (F or R or T ), work.<br>done --> (C or I or D).<br>comment --> (E or blank). | Final stage of a call, which may include a tentative fix, one of several symbols indicating that work is done, and a trailing comment that explains what was done.<br>Note that there are two kinds of closings (with and without a "tentative fix", and that each constituent in the closing (tentative fix, done, and comment) has its own rule. |
| work --> (W, W, . . . ). | An episode of work on the call, consisting of a string of "w"s of any length. |
| user --> US, PS.<br>user --> US, work, PS. | Change responsibility for a call to the user and back to product support, with an optional work episode intervening. |
| developer --> DV, PS.<br>developer --> DV, work, PS | Escalate a call to a development and then back to product support, with an optional work episode intervening. |

**Figure 5: Rewriting a sample call**

| Rewrite Rule Applied | Resulting String of symbols | | | | | | |
|---|---|---|---|---|---|---|---|
| (Rawdata) | O | W | W | US | PS | C | E |
| Opening-->O. | Opening | W | W | US | PS | C | E |
| Done --> ( C or I or D ). | Opening | W | W | US | PS | Done | E |
| Comment --> (E or blank). | Opening | W | W | US | PS | Done | Comment |
| Closing --> Done, Comment. | Opening | W | W | US | PS | Closing | |
| Work --> ( W, W, ...). | Opening | Work | | US | PS | Closing | |
| User --> US, PS. | Opening | Work | | User | | Closing | |
| Search for solution --> any order( Prioritize, Transfer, Work, User, Developer). | Opening | Search for solution | | | | Closing | |
| CALL --> Opening, Search for solution, Closing. | CALL | | | | | | |